FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 31, N^o 1, March 2018, pp. 25 - 39
https://doi.org/10.2298/FUEE1801025B

COMPARATIVE EVALUATION OF QUASI-DELAY-INSENSITIVE ASYNCHRONOUS ADDERS CORRESPONDING TO RETURN-TO-ZERO AND RETURN-TO-ONE HANDSHAKING

Padmanabhan Balasubramanian

School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

Abstract. *This article makes a comparative evaluation of quasi-delay-insensitive (QDI) asynchronous adders, realized using the delay-insensitive dual-rail code, which adhere to 4-phase return-to-zero (RTZ) and 4-phase return-to-one (RTO) handshake protocols. The QDI adders realized correspond to the following adder architectures: i) ripple carry adder, ii) carry lookahead adder, and iii) carry select adder. The QDI adders correspond to three different timing regimes viz. strong-indication, weak-indication, and early output. They are physically implemented using a 32/28nm CMOS process. The comparative evaluation shows that, overall, QDI adders which correspond to the 4-phase RTO handshake protocol are better than the QDI adder counterparts which correspond to the 4-phase RTZ handshake protocol in terms of latency, area, and average power dissipation.*

Key words: *asynchronous circuits, QDI, adders, indication, standard cells, CMOS*

1. INTRODUCTION

The International Technology Roadmap for Semiconductors (ITRS 2.0) [1] has identified design for variability as one of the key challenges for nanoelectronics. Process variability and device variability have assumed more significance in the nanoelectronics era compared to the microelectronics era. This is because random dopant and atomistic fluctuations, high heat flux, negative bias temperature instability, electro-migration, hot carrier effects, stress-induced variation, process-induced defects, electrostatic discharge, and metrology and other manufacturing issues have become more prominent in the nanoelectronics era compared to the microelectronics era. To overcome these issues, solutions are being developed at various levels such as at material-level, process-level, device-level, circuit-level, and the system-level [2].

Received September 18, 2017
Corresponding author: Padmanabhan Balasubramanian
The author is now with the School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798
(E-mail: balasubramanian@ntu.edu.sg)

At the circuit-level, the QDI[1] asynchronous design method [3] employing delay-insensitive code(s) for data representation and processing and a 4-phase handshake protocol for data communication is considered to be robust and is construed to be a viable alternative to the synchronous design method [4]. This is because QDI circuits encompass several advantages [5] such as low power [6 – 9], tolerance to noise and electromagnetic interference [10 – 12], ability to withstand process, voltage and temperature variations [13] [14], self-checking [15], resistant to side channel attacks in the case of secure applications [16 – 19] etc.

In general, QDI circuits widely employ the delay-insensitive dual-rail data encoding and the 4-phase RTZ handshaking [20]. However, a new 4-phase RTO handshake protocol was proposed [21] for QDI circuits. Based on a few case studies [22] [23], it was reported that QDI circuits which correspond to the RTO protocol report better design metrics than their QDI circuit counterparts adhering to the RTZ protocol. QDI circuits performing data transactions based on either the RTZ or the RTO protocol are robust. QDI circuits and systems are guaranteed to be correct by construction since they adopt unbounded delay models for gates and wires, with the only exception of isochronic forks[2] [24] which represent the weakest compromise to delay-insensitivity.

In this work, the adder which forms an important datapath of any processing unit is considered for the analysis to compare the efficiency of the RTO protocol versus the RTZ protocol. Various adder architectures such as the ripple carry adder (RCA), the carry lookahead adder (CLA), and the carry select adder (CSLA) are considered for QDI implementation based on the RTZ and RTO protocols to perform a comprehensive comparative evaluation. This work builds upon [25], wherein only the RCA architecture was considered to comparatively evaluate the RTZ and RTO protocols.

The rest of this article is organized as follows. Section 2 provides an overview of: i) QDI circuit operation encompassing delay-insensitive data encoding and data transaction using the RTZ and RTO handshake protocols, and ii) the types of QDI circuits, their timing characteristics, and their general properties. Section 3 presents the logic rules for transforming QDI circuits corresponding to the RTZ protocol into QDI circuits adhering to the RTO protocol and vice-versa. Also, some circuit illustrations are provided in this section. Section 4 presents the simulation results corresponding to several 32-bit QDI RCAs, CLAs, and CSLAs, implemented using delay-insensitive dual-rail data encoding and adhering to RTZ and RTO handshaking. The QDI adders realized correspond to strong-indication, weak-indication, and early output. Section 5 provides the conclusions.

2. QDI Circuit Operation, Types and Properties

2.1. Operation of QDI Circuit

The architecture of a QDI circuit is correlated with the sender (SX) and receiver (RX) analogy in Figure 1a. The QDI circuit is sandwiched between the current stage and the

[1] QDI design represents a robust flavor of asynchronous circuit design methods. QDI circuits are the practically realizable delay-insensitive asynchronous circuits.

[2] An isochronic fork implies that the up-going or down-going signal transitions on all the ends of the fork are assumed to be concurrent.

next stage register banks. A register in a QDI design is a 2-input C-element that is represented by the circle with the marking C in the figures. The C-element outputs binary 1 or 0 only if all its inputs are binary 1 or 0 respectively and would maintain its existing steady-state even if any of its inputs is different.

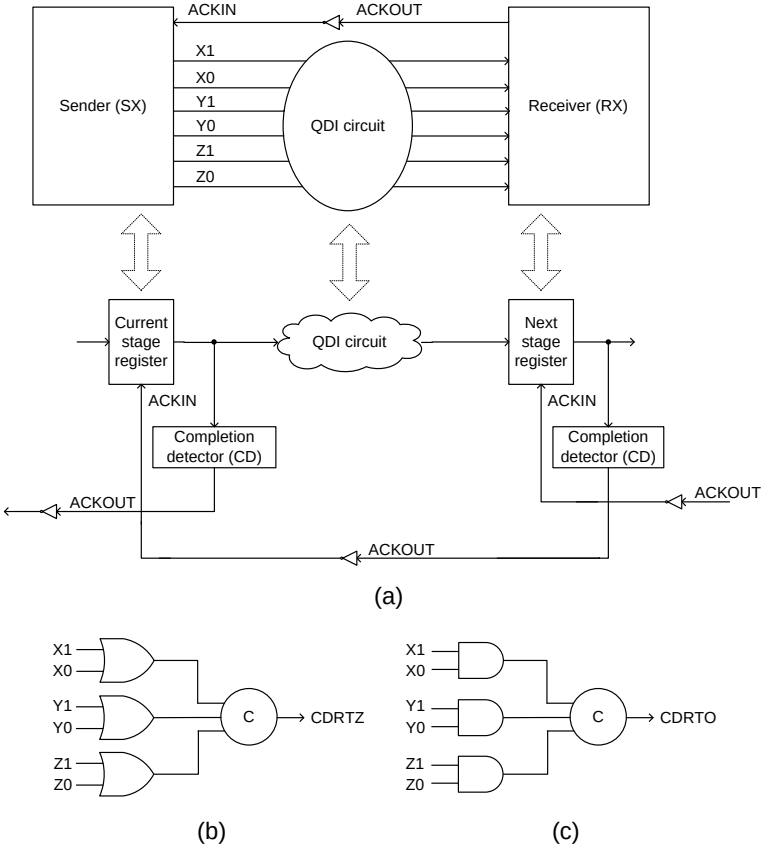

Fig. 1 (a) an asynchronous circuit, correlated with the sender-receiver analogy for illustration. Completion detectors corresponding to (b) the RTZ handshake protocol, and (c) the RTO handshake protocol.

A single-rail data wire X is encoded using the dual-rail code [26] into two data wires as X1 and X0. Based on the RTZ protocol [20], the data X = 1 is represented by X1 = 1 and X0 = 0, and the data X = 0 is represented by X0 = 1 and X1 = 0. X1 = X0 = 0 represents the spacer. X1 = X0 = 1 is invalid since the coding scheme is unordered [27] and where no code word is allowed to be a subset of another code word. According to the RTZ protocol, the application of primary inputs to a QDI circuit should follow the sequence: data-spacer-data-spacer, and so forth, with each input data followed by the RTZ of the encoded data wires. Note that binary 1 is used to represent data with respect to the RTZ protocol. On the other hand, according to the RTO protocol [21], binary 0 is used to

represent data. As per the RTO protocol, the valid data Y = 1 is represented by Y1 = 0 and Y0 = 1, and Y = 0 is represented by Y0 = 0 and Y1 = 1. The spacer is represented by Y0 = Y1 = 1. Y1 = Y0 = 0 is deemed invalid since the coding scheme is unordered. As per the RTO protocol, the application of primary inputs to a QDI circuit follows the sequence: spacer-data-spacer-data, and so forth, with each input data followed by the RTO of the encoded data wires.

The 4-phase handshake protocol, whether it is RTZ or RTO, consists of four phases which will be explained with reference to Figure 1a by considering dual-rail encoded data. However, the explanation would be applicable for data represented using any delay-insensitive 1-of-*n* code [26]. As per the RTZ protocol, in the first phase, the dual-rail data bus shown in Figure 1a which is specified by (X1, X0), (Y1, Y0), and (Z1, Z0) is in the spacer state, and ACKIN is high. SX transmits data and this results in rising signal transitions on anyone of the corresponding dual rails of the entire dual-rail data bus. In the second phase, RX receives the data sent, and it drives ACKOUT high. In the third phase, SX waits for ACKIN to go low and then resets the entire dual-rail data bus to the spacer state i.e. all 0s. In the fourth phase, after an unbounded but a finite and positive time, RX would drive ACKOUT low i.e. ACKIN becomes high. With this one data transaction is said to be complete, and the asynchronous circuit is ready to proceed with the next data transaction. An example completion detector, which comprises the dual-rail encoded primary inputs (X1, X0), (Y1, Y0), and (Z1, Z0), that indicates or acknowledges the receipt of data and the all zeroes spacer on the primary inputs through its output CDRTZ is illustrated in Figure 1b. The completion detector shown in Figure 1b corresponds to the RTZ protocol.

With respect to the RTO handshake protocol, in the first phase, ACKIN is 1. SX would transmit the spacer i.e. all 1s, and this causes rising signal transitions on all the rails of the dual-rail data bus. In the second phase, RX receives the spacer sent, and it drives ACKOUT high. In the third phase, TX waits for ACKIN to assume 0 and then sends the input data by resetting any one of the corresponding dual-rails of the entire dual-rail data bus. Then in the fourth phase, after an unbounded but a finite and positive time, RX would drive ACKOUT low i.e. ACKIN becomes high. With this one data transaction is said to be complete, and the QDI circuit is ready to commence the next data transaction. An example completion detector that comprises the dual-rail encoded primary inputs (X1, X0), (Y1, Y0), and (Z1, Z0), which indicates the receipt of data and the all ones spacer on the primary inputs through its output CDRTO is depicted by Figure 1c. This completion detector corresponds to the RTO protocol.

2.2. Types of QDI Circuits

QDI circuits are classified as strongly indicating, weakly indicating, and early output types [28]. A strong-indication QDI circuit [29] [30] waits to receive all the primary inputs, whether they are data or spacer, and then starts data processing to produce the required primary outputs. A weak-indication QDI circuit [29] [31] would produce some of the primary outputs after receiving a subset of the primary inputs. However, the production of at least one primary output is delayed till the last primary input is received. An early output QDI circuit [32] [33] is the most relaxed of the three in that it is able to produce all the primary outputs after receiving a subset of the primary inputs. If an early output QDI circuit produces data early, it is said to be of early set type, and if an early output QDI circuit assumes the spacer state early, it is said to be of early reset type. The

input-output timing behaviour of strong-indication, weak-indication, and early output QDI circuits is captured by Figure 2. The early set and reset behaviours are shown in Figure 2.

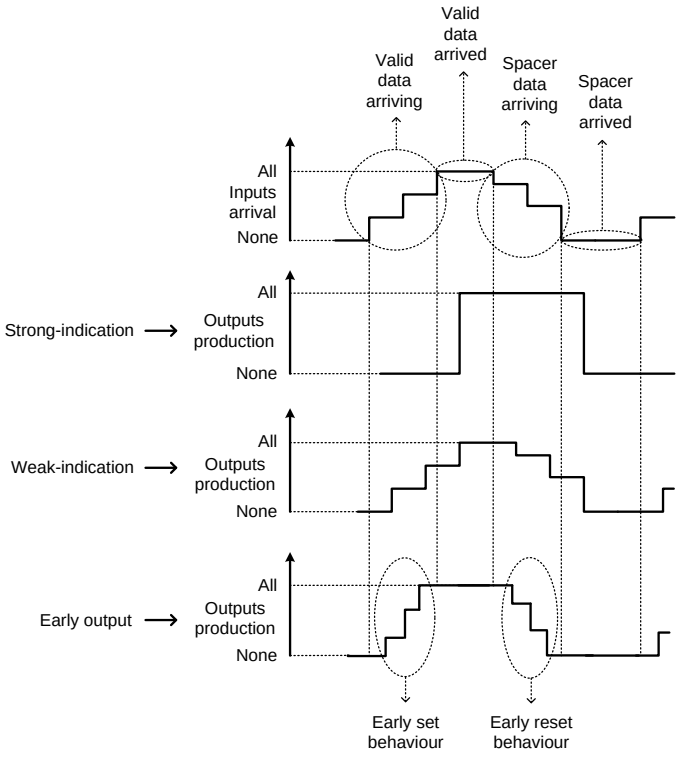

Fig. 2 Input-output timing characteristic of strong-indication, weak-indication, and early output QDI circuits

2.3. General Properties of QDI Circuits

QDI circuits, regardless of whether they are strongly indicating or weakly indicating or early output type, have some properties in common. Firstly, QDI circuits should be free of wire and gate orphans [34] [35]. A wire orphan refers to an unacknowledged signal transition on a wire. The wire orphan problem, if any, can be resolved through the isochronic fork assumption. A gate orphan is an unacknowledged signal transition on an intermediate gate output. The gate orphan problem is difficult to resolve and to overcome it, sophisticated timing assumption(s) might be required. Secondly, QDI circuits tend to satisfy the monotonic cover constraint [16], which implies the activation of a unique signal path from a primary input to a primary output for each input data applied. The monotonic cover constraint is implicit in a disjoint sum-of-products expression [36], which is used to synthesize a QDI circuit. In a disjoint sum-of-products expression, the product terms are mutually orthogonal, i.e. the logical conjunction of any two product terms in a disjoint sum-of-products expression yields null [37 – 39]. Thirdly, the signal

transitions ripple monotonically [40] from the first logic level up to the last logic level in a QDI circuit [41]. The transitions either increase or decrease monotonically. For a QDI circuit that adheres to the RTZ protocol, for the application of data, the transitions would increase monotonically and for the application of spacer, the transitions would decrease monotonically. On the contrary, for a QDI circuit adhering to the RTO protocol, for the application of spacer, the transitions would increase monotonically, and for the application of data, the transitions would decrease monotonically throughout the circuit.

It is important to ascertain the type of a QDI circuit when it is composed using many QDI sub-circuits, as is common in the design of QDI arithmetic circuits. In general, a cascade of strong-indication or weak-indication or early output QDI sub-circuits yields a strong-indication or a weak-indication or an early output QDI circuit respectively. Sometimes there might be an exception when composing early output QDI sub-circuits. For example, it was noted in [42] [43] that a cascade of early output QDI full adders led to a relative-timed RCA, whereas in [33] [44] a cascade of early output QDI full adders led to an early output RCA. This might be because in terms of robustness, the strong-indication timing model tops the hierarchy followed by the weak-indication timing model, which is succeeded by the early output timing model. The relative-timing model is not QDI and is the least robust of the asynchronous timing models described. Relative-timed asynchronous circuits [45] require explicit and perhaps complicated timing assumptions to ensure their safe operation but could exhibit more optimized design metrics compared to the QDI circuits. Hence, in the case of relative-timing, the robustness is traded off for greater design optimization [46]. Further, a cascade of QDI sub-circuits with more robust and less robust timing models generally causes the least robust timing model to be ascribed to the resultant QDI circuit. For example, a cascade of strong-indication and weak-indication QDI sub-circuits leads to a weak-indication QDI circuit. A cascade of strong-indication and/or weak-indication QDI sub-circuits and early output QDI sub-circuit(s) leads to an early output QDI circuit.

3. Logic Rules for RTZ to RTO and Vice-Versa Protocol Conversion

QDI circuits, regardless of whether they correspond to the RTZ or the RTO protocol, when physically realized, generally consist of C-elements[3] and simple and complex logic gates. Any C-elements used in a QDI circuit, whether they correspond to the RTZ or the RTO protocol, would remain unchanged and their inputs would also be unchanged when transforming a QDI circuit which adheres to the RTZ protocol into a QDI circuit which corresponds to the RTO protocol and vice-versa. The logic transformation rules to be discussed below, which could facilitate the RTZ to RTO and vice-versa protocol conversion are applicable only to the discrete and complex logic gates comprising the respective circuits and excludes any C-elements. The logic transformation rules for the handshake protocols conversion tend to obey the well-established duality principle of Boolean algebra. The duality principle [47] states that every algebraic expression that is deduced using the postulates of Boolean algebra remains valid if the logical operators and

[3] The C-element outputs binary 1 or 0 only when all its inputs are binary 1 or 0. If any of its inputs is different, the C-element would maintain its existing steady-state. The C-element is portrayed by an AND gate with the marking ‘C’ on its periphery.

identity elements are interchanged. Herein, it implies that for the RTZ to RTO protocol conversion the AND operator should be replaced by the OR operator and the OR operator should be replaced by the AND operator; the reverse is applicable for the RTO to RTZ protocol conversion. An example set of logic transformation rules for the handshake protocols conversion and their proofs by induction are provided below. These rules may be extended without any loss of generality depending upon a QDI circuit composition.

$$\text{RTZ: } P + Q \leftrightarrow \text{RTO: } PQ \tag{1}$$

$$\text{RTZ: } P + QR \leftrightarrow \text{RTO: } P\,(Q + R) \tag{2}$$

$$\text{RTZ: } PQ + RS \leftrightarrow \text{RTO: } (P + Q)\,(R + S) \tag{3}$$

The function (P + Q) corresponding to the RTZ protocol, given in (1), is implemented using a 2-input OR gate, and the RTO equivalent viz. PQ is implemented using a 2-input AND gate. Table 1 shows the proof by induction for (1). The 2-input OR and AND gates are simple logic gates present in a standard digital cell library [48]. The function (P + QR) corresponding to the RTZ protocol, given in (2), can be implemented using the AO21 gate and its RTO equivalent viz. P (Q + R) can be implemented using the OA21 gate. Table 2 shows the proof by induction for (2). The function (PQ + RS) corresponding to the RTZ protocol, given in (3), can be implemented using the AO22 gate and the RTO equivalent i.e. (P + Q) (R + S) can be implemented using the OA22 gate. Table 3 shows the proof by induction for (3). The AO21, OA21, AO22 and OA22 gates are complex logic gates present in a standard digital cell library [48].

Table 1 Proof by induction for (1)

Inputs		RTZ	RTO
P	Q	P + Q	PQ
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

Recall that binary 1 is used to represent the data with respect to the RTZ protocol and binary 0 is used to represent the data with respect to the RTO protocol after data encoding. This is conformance with the duality property of Boolean algebra, which states that identity elements can be interchanged [47]. As mentioned in Section 2.1, the zeroes spacer is used in the case of the RTZ protocol and the ones spacer is used in the case of the RTO protocol. Given these, it can be seen from Table 1 that if the input P or Q is 1, which indicates the data with respect to the RTZ protocol, (P + Q) would yield 1, and when P and Q are 0 then (P + Q) would yield 0 indicating the RTZ state. On the other hand, if either P or Q is 0 in Table 1, which indicates the data based on the RTO protocol, PQ would evaluate to 0, and when P and Q are 1, PQ would evaluate to 1, which indicates the RTO state.

In Tables 2 and 3, sub-functions are additionally introduced for the sakes of clarity and illustration. In the case of Table 2, if P or QR is 1, then (P + QR) evaluates to 1 signifying the data according to the RTZ protocol, and if P and QR are 0, then (P + QR)

evaluates to 0 signifying the RTZ state. If P or (Q + R) is 0, then P (Q + R) evaluates to 0 signifying the data according to the RTO protocol, and if P and (Q + R) are 1, then P (Q + R) evaluates to 1 signifying the RTO state. With respect to Table 3, if PQ or RS is 1, then (PQ + RS) evaluates to 1 signifying the data as per the RTZ protocol, and if PQ and RS are 0, then (PQ + RS) evaluates to 0 signifying the RTZ state. However, if (P + Q) or (R + S) is 0, then (P + Q) (R + S) evaluates to 0 signifying the data as per the RTO protocol. Supposing (P + Q) and (R + S) are 1, then (P + Q) (R + S) would evaluate to 1 signifying the RTO state.

Table 2 Proof by induction for (2)

Inputs			RTZ sub-function	RTZ	RTO sub-function	RTO
P	Q	R	QR	P + QR	Q + R	P (Q + R)
0	0	0	0	0	0	0
0	0	1	0	0	1	0
0	1	0	0	0	1	0
0	1	1	1	1	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	1
1	1	0	0	1	1	1
1	1	1	1	1	1	1

Table 3 Proof by induction for (3)

Inputs				RTZ sub-functions		RTZ	RTO sub-functions		RTO
P	Q	R	S	PQ	RS	PQ + RS	P + Q	R + S	(P + Q) (R + S)
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1	0
0	0	1	0	0	0	0	0	1	0
0	0	1	1	0	1	1	0	1	0
0	1	0	0	0	0	0	1	0	0
0	1	0	1	0	0	0	1	1	1
0	1	1	0	0	0	0	1	1	1
0	1	1	1	0	1	1	1	1	1
1	0	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	1	1	1
1	0	1	0	0	0	0	1	1	1
1	0	1	1	0	1	1	1	1	1
1	1	0	0	1	0	1	1	0	0
1	1	0	1	1	0	1	1	1	1
1	1	1	0	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1

Example circuits to illustrate the conversion from RTZ to RTO protocol and vice-versa are shown in Figure 3.

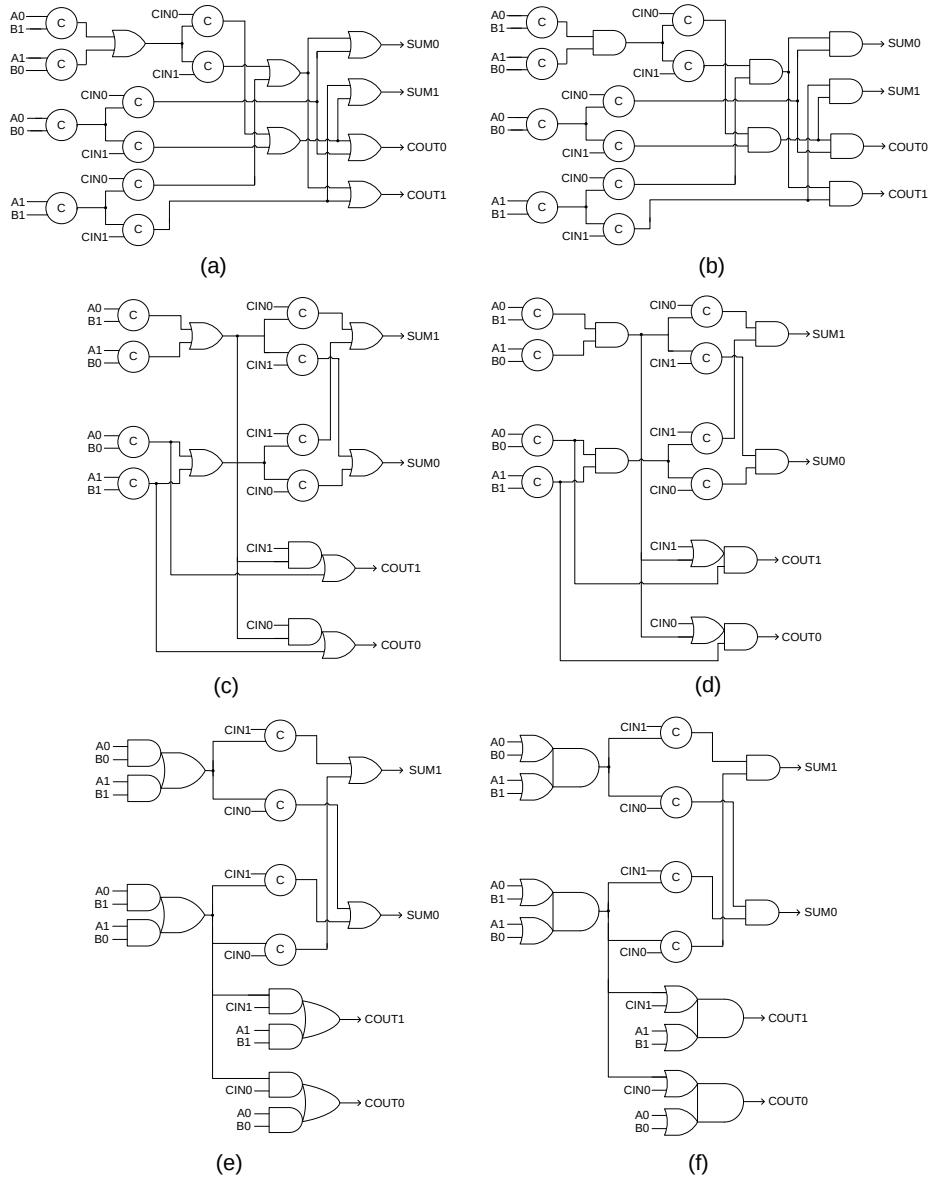

Fig 3 Strongly indicating full adder [50] corresponding to (a) RTZ handshaking and (b) RTO handshaking; Weakly indicating full adder [51] corresponding to (c) RTZ handshaking and (d) RTO handshaking; Early output full adder [33], corresponding to (e) RTZ handshaking (early reset type) and (f) RTO handshaking (early set type)

Figure 3 portrays strong-indication, weak-indication and early output implementations of the full adder. The full adder adds an augend and an addend along with a carry input

and produces the sum output and any carry overflow. In Figure 3, (A1, A0), (B1, B0) and (CIN1, CIN0) represent the dual-rail augend, addend and carry inputs of the full adders, and (SUM1, SUM0) and (COUT1, COUT0) represent the dual-rail sum and carry outputs. Figures 3a, 3c and 3e depict full adder implementations which correspond to the RTZ protocol, and Figures 3b, 3d and 3f show the respective full adder realizations which correspond to the RTO protocol. The 2-input OR gates, AO21 gates and AO22 gates of Figures 3a, 3c and 3e are replaced by 2-input AND gates, OA21 gates and OA22 gates respectively in Figures 3b, 3d and 3f, in accordance with (1), (2) and (3), given earlier. Note that there is no change whatsoever in the inputs or outputs of the corresponding circuits belonging to the RTZ and the RTO protocols in Figure 3. Moreover, the 2-input C-elements and their corresponding inputs remain unchanged.

4. SIMULATION RESULTS

Several 32-bit QDI RCAs [49 – 55], CLAs [56] [57] and CSLAs [58] were semi-custom realized using the standard digital library cells of a 32/28nm CMOS process [48]. The 2-input C-element was alone custom realized by modifying the AO222 complex gate by introducing feedback. The 2-input C-element was realized using 12 transistors and was made available to implement the various QDI adders, which correspond to RTZ and RTO protocols. Any high-input C-element functionality, wherever likely in an adder design, was safely decomposed in QDI style [59] to avoid the problem of gate orphans.

About 1000 random input vectors were identically supplied to all the QDI adders through a test bench at time intervals of 20ns to perform the functional simulations and to capture their respective switching activities. The value change dump (.vcd) files generated through the functional simulations were used to estimate the average power dissipation. The worst-case (forward) latency, i.e. the critical path delay and the area of the QDI adders were also estimated. A default wire load model was considered while estimating the design metrics to include the effect of parasitic in the simulations. The design metrics viz. latency, area, and average power dissipation, estimated for the various QDI adders, which correspond to the RTZ and RTO protocols are given in Table 4. The input registers and the completion detectors of the various QDI adders corresponding to the RTZ and RTO protocols are respectively identical. So the differences between their design metrics can be attributed to the respective differences between their function blocks.

Before discussing the results, it should be noted that the focus of this article is not to comment on the efficiency of various adder architectures or about the type of the adders with relation to latency or area or power optimization, and these have already been discussed in the published literature. Rather, the intent of this article is to provide a comparison between the design metrics of different QDI adders based on their realization using RTZ and RTO protocols, and eventually to arrive at a general conclusion regarding which of these handshake protocols is more preferable to potentially achieve enhanced optimizations in the design metrics regardless of the extent of optimization achievable. The improvements in the design metrics which may be achieved by one protocol over the other could in part be explained as due to the differences in the implementation. However, the extent of optimizations in the design metrics achievable would also depend on the digital cell library targeted, the technology node and the PVT corner chosen to perform

the simulations. Thus the results given in Table 4 are to be used as a reference to guide the choice of a 4-phase handshake protocol for the effective design of QDI circuits.

Table 4 Design metrics of 32-bit QDI adders corresponding to RTZ and RTO protocols, estimated using Synopsys tools based on implementation using a 32/28nm CMOS process

QDI adder reference; and adder type	4-phase RTZ handshake protocol			4-phase RTO handshake protocol		
	Latency (ns)	Area (μm^2)	Power (μW)	Latency (ns)	Area (μm^2)	Power (μW)
RCAs						
[49]; SI	14.61	2529	2190	14.15	2529	2185
[52]; SI	9.26	2504.60	2181	8.74	2374.48	2167
[50]; SI	9.04	2293.14	2172	8.88	2293.15	2168
[52]; WI	8.24	2423.27	2177	8.03	2358.21	2167
[53]; WI	7	2016.63	2171	6.95	2016.63	2167
[54]; WI	9.66	2642.85	2192	9.66	2642.85	2191
[55]; WI	4.43	2097.96	2174	3.79	2097.96	2170
[51]; WI	3.32	2049.16	2171	3.31	2049.16	2167
[33]; EO	3.10	1658.80	2161	2.93	1658.80	2157
[44]; EO	2.14	2436.48	2173	2.13	2649.96	2176
CLAs						
[56], [55]; WI: Regular	3.31	2951.88	2191	3.19	2984.41	2184
[56], [55]; WI: Hybrid	3.08	2845.14	2189	2.97	2873.61	2182
[56], [55]; WI: Regular with alias logic	2.46	2992.55	2192	2.36	3025.08	2185
[56], [55]; WI: Hybrid with alias logic	2.38	2880.72	2190	2.29	2909.19	2183
[56], [51]; WI: Regular	3.14	2915.29	2188	3.10	2947.82	2182
[56], [51]; WI: Hybrid	2.93	2807.02	2186	2.89	2835.49	2180
[56], [51]; WI: Regular with alias logic	2.32	2955.95	2190	2.30	2988.48	2183
[56], [51]; WI: Hybrid with alias logic	2.25	2842.60	2187	2.22	2871.07	2181
[57]; EO: Regular	2.75	2569.65	2177	2.73	2553.39	2169
[57]; EO: Hybrid	2.53	2455.80	2175	2.51	2441.56	2167
CSLAs						
[58] – [33], [60]; EO: Non-uniform	3.23	3384.44	2312	3.15	3384.44	2303
[58] – [33], [60]; EO: Uniform	2.46	3000.17	2293	2.38	3000.17	2285

Legends used: SI – Strong-indication; WI – Weak-indication; EO – Early output
Hybrid CLAs incorporate a 4-bit least significant RCA, which improves the design metrics of Regular CLAs

Overall, it can be observed from Table 4 that the QDI adders based on the RTO protocol feature less latency (and hence less cycle time) and power dissipation and occupy almost the same area than their QDI adder counterparts based on the RTZ protocol. The completion detector of a QDI circuit corresponding to the RTZ protocol consists of a series of 2-input OR gates whose outputs are synchronized by a C-element tree. On the other hand, the completion detector of a QDI circuit adhering to the RTO

protocol comprises a series of 2-input AND gates whose outputs are synchronized by a tree of C-elements. Further, any 2-input OR gates present in the functional block(s) of a QDI adder corresponding to the RTZ protocol would be replaced by 2-input AND gates in the functional block(s) of a QDI adder counterpart adhering to the RTO protocol. In static CMOS implementations, it is well known that the OR gate is more expensive than the AND gate in terms of delay, area, and power dissipation [61] due to the series stacking of pMOS transistors in the pull-up network of the former contrary to the parallel stacking of pMOS transistors in the pull-up network of the latter. Hence the use of 2-input AND gates instead of 2-input OR gates in the QDI adders and their respective completion detectors implies better optimized design metrics can be expected for the RTO protocol compared to the RTZ protocol.

In Table 4, it can be noticed that in some scenarios the areas of the QDI adders corresponding to the RTZ and RTO protocols are the same. For examples, the non-uniform 32-bit CSLA with the input partition of 8-7-6-4-3-2-2 and the uniform 32-bit CSLA with the input partition of 8-8-8-8 occupy similar areas with respect to both the handshake protocols. The non-uniform and uniform QDI CSLAs, highlighted in Table 4, are constructed using the early output full adder of [33], and the strongly indicating 2:1 multiplexer (MUX) of [60]. With respect to the RTZ protocol, the early output full adder of [33] consists of four AO22 gates, four 2-input C-elements and two 2-input OR gates, as shown in Figure 3e. Based on the RTO protocol, the early output full adder of [33] would comprise four OA22 gates, four 2-input C-elements and two 2-input AND gates, as shown in Figure 3f. The strongly indicating 2:1 MUX design of [60], which is called SIDCO, requires seven 2-input C-elements and four 2-input OR gates for realization based on the RTZ protocol. On the other hand, for implementation based on the RTO protocol, the strongly indicating 2:1 MUX design would require seven 2-input C-elements and four 2-input AND gates. The AO22 and OA22 gates of the digital cell library [48] have the same area of 2.54μm^2, and the 2-input OR gate and the 2-input AND gate occupy the same area of 2.03μm^2. As a result, the areas of the full adder and the 2:1 MUX of a QDI CSLA would be the same regardless of the handshake protocol adopted. This explains why the non-uniform and uniform QDI CSLAs in Table 4 feature the same area with respect to both RTZ and RTO protocols. Although the areas of AO22 and OA22 gates, and the areas of the 2-input OR gate and the 2-input AND gate are the same in [48], their corresponding delay and power dissipation values are different. This is the reason why the QDI CSLAs based on the RTO protocol have less latency and power dissipation than the QDI CSLAs based on the RTZ protocol, as seen in Table 4.

Having similar cell areas for the dual logic gates viz. OR and AND, AO21 and OA21, AO22 and OA22 etc. in [48] is rather uncommon in the case of commercial standard cell libraries. The standard digital cell library [48] does not have foundry support and is meant for use for academic teaching and research. Hence, it may be safely hypothesized that if a commercial digital cell library is used for the physical implementation of the QDI adders given in Table 4, then the RTO protocol would facilitate higher percentage optimizations in the design metrics than the RTZ protocol and therefore the improvements in the design metrics reported in Table 4 would tend to serve as a baseline.

5. CONCLUSIONS

This article discussed the implementation of various QDI adders, which correspond to diverse architectures and timing regimes by utilizing the delay-insensitive dual-rail code, based on the 4-phase RTZ and RTO handshake protocols. The logic transformation rules governing the circuit conversions between RTZ and RTO protocols were presented, and their proofs by induction were also provided. The simulations were performed by using a 32/28nm CMOS process. The simulation results show that QDI adders corresponding to the RTO protocol generically feature improved design parameters than the QDI adder counterparts which adhere to the RTZ protocol. Hence it is concluded that the 4-phase RTO protocol is potentially more efficient than the 4-phase RTZ protocol to implement handshaking in QDI asynchronous (arithmetic) circuits.

REFERENCES

[1] ITRS design report. Available: http://www.itrs2.net

[2] S. Kundu and A. Sreedhar, *Nanoscale CMOS VLSI Circuits: Design for Manufacturability*, McGraw-Hill, New York, USA, 2010.

[3] A.J. Martin, S.M. Burns, T.K. Lee, D. Borkovic and P.J. Hazewindus, "The first asynchronous microprocessor: the test results," *ACM SIGARCH Computer Architecture News*, vol. 17, pp. 95-98, 1989.

[4] A.J. Martin and M. Nystrom, "Asynchronous techniques for system-on-chip design," *Proceedings of the IEEE*, vol. 94, pp. 1089-1120, 2006.

[5] C.H. Van Kees Berkel, M.B. Josephs and S.M. Nowick, "Scanning the technology applications of asynchronous circuits", *Proceedings of the IEEE*, vol. 87, pp. 223-233, 1999.

[6] S.B. Furber, D.A. Edwards and J.D. Garside, "AMULET3: a 100 MIPS asynchronous embedded processor," In Proceedings of the International Conference on Computer Design, pp. 329-334, 2000.

[7] L. Necchi, L. Lavagno, D. Pandini and L. Vanzago, "An ultra-low energy asynchronous processor for wireless sensor networks," In Proceedings of the 12th IEEE International Symposium on Asynchronous Circuits and Systems, 2006, pp. 1-8.

[8] B.Z. Tang and F. Lane, "Low power QDI asynchronous FFT," In Proceedings of the 22nd IEEE International Symposium on Asynchronous Circuits and Systems, 2016, pp. 87-88.

[9] W. Jiang, D. Bertozzi, G. Miorandi, S.M. Nowick, W. Burleson and G. Sadowski, "An asynchronous NoC router in a 14nm FinFET library: comparison to an industrial synchronous counterpart," In Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, 2017, pp. 732-733.

[10] N.C. Paver, P. Day, C. Farnsworth, D.L. Jackson, W.A. Lien and J. Liu, "A low-power, low noise, configurable self-timed DSP", In Proceedings of the 4th International Symposium on Advanced Research in Asynchronous Circuits and Systems, pp. 32-42, 1998.

[11] A.J. Martin and M. Nystrom, "Asynchronous techniques for noise tolerant nanoelectronics," Technical Report Situs-TR-04-01, Situs Logic, Pasadena, CA, USA, 2004.

[12] G.F. Bouesse, G. Sicard, A. Baixas and M. Renaudin, "Quasi delay insensitive asynchronous circuits for low EMI", In Proceedings of the 4th International Workshop on Electromagnetic Compatibility of Integrated Circuits, 2004, pp. 27-31.

[13] K.J. Kulikowski, V. Venkataraman, Z. Wang, A. Taubin and M. Karpovsky, "Asynchronous balanced gates tolerant to interconnect variability", In Proceedings of the IEEE International Symposium on Circuits and Systems, 2008, pp. 3190-3193.

[14] I.J. Chang, S.P. Park and K. Roy, "Exploring asynchronous design techniques for process-tolerant and energy-efficient subthreshold operation", *IEEE Journal of Solid-State Circuits*, vol. 45, pp. 401-410, 2010.

[15] I. David, R. Ginosar and M. Yoeli, "Self-timed is self-checking", *Journal of Electronic Testing: Theory and Applications*, vol. 6, pp. 219-228, 1995.

[16] L.A. Plana, P.A. Riocreux, W.J. Bainbridge, A. Bardsley, S. Temple, J.D. Garside, Z.C. Yu, "SPA – a secure Amulet core for smartcard applications," *Microprocessors and Microsystems*, vol. 27, pp. 431-446, 2003.

[17] D. Sokolov, J. Murphy, A. Bystrov and A. Yakovlev, "Design and analysis of dual-rail circuits for security applications", *IEEE Transactions on Computers*, vol. 54, pp. 449-460, 2005.
[18] F. Burns, A. Bystrov, A. Koelmans and A. Yakovlev, "Design and security evaluation of balanced 1-of-n circuits," *IET Computers and Digital Techniques*, vol. 6, pp. 125-135, 2012.
[19] W. Cilio, M. Linder, C. Porter, J. Di, D.R. Thompson and S.C. Smith, "Mitigating power- and timing-based side-channel attacks using dual-spacer dual-rail delay-insensitive asynchronous logic," *Microelectronics Journal*, vol. 44, pp. 258-269, 2013.
[20] J. Sparsø and S. Furber (Eds.), *Principles of Asynchronous Circuit Design: A Systems Perspective*, Kluwer Academic Publishers, 2001.
[21] M.T. Moreira and N.L.V. Calazans, "Quasi-delay-insensitive return-to-one design," In Proceedings of the Design, Automation and Test in Europe Conference and Exhibition PhD Forum, 2014, pp. 1-2.
[22] M.T. Moreira, J.J.H. Pontes and N.L.V. Calazans, "Tradeoffs between RTO and RTZ in WCHB QDI asynchronous design," In Proceedings of the 15th International Symposium on Quality Electronic Design, 2014, pp. 692-699.
[23] R.A. Guazzelli, M.T. Moreira and N.L.V. Calazans, "A comparison of asynchronous QDI templates using static logic," In Proceedings of the 8th IEEE Latin American Symposium on Circuits and Systems, 2017, pp. 1-4.
[24] A.J. Martin, "The limitation to delay-insensitivity in asynchronous circuits," In Proceedings of the 6th MIT Conference on Advanced Research in VLSI, 1990, pp. 263-278.
[25] P. Balasubramanian, C. Dang, "A comparison of quasi-delay-insensitive asynchronous adder designs corresponding to return-to-zero and return-to-one handshaking," In Proceedings of the 60th IEEE International Midwest Symposium on Circuits and Systems, 2017, pp. 1192-1195.
[26] T. Verhoeff, "Delay-insensitive codes – an overview", *Distributed Computing*, vol. 3, pp. 1-8, 1988.
[27] B. Bose, "On unordered codes", *IEEE Transactions on Computers*, vol. 40, pp. 1-8, 1988.
[28] P. Balasubramanian, "Comments on "Dual-rail asynchronous logic multi-level implementation"," *Integration, the VLSI Journal*, vol. 52, pp. 34-40, 2016.
[29] C.L. Seitz, "System Timing", in *Introduction to VLSI Systems*, C. Mead and L. Conway (Editors), pp. 218-262, Addison-Wesley, Reading, Massachusetts, USA, 1980.
[30] P. Balasubramanian and D.A. Edwards, "Efficient realization of strongly indicating function blocks", In Proceedings of the IEEE Computer Society Annual Symposium on VLSI, 2008, pp. 429-432.
[31] P. Balasubramanian and D.A. Edwards, "A new design technique for weakly indicating function blocks", In Proceedings of the 11th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems, 2008, pp. 116-121.
[32] C. Brej, "Early output logic and anti-tokens," PhD thesis, School of Computer Science, The University of Manchester, 2006.
[33] P. Balasubramanian, "A robust asynchronous early output full adder," *WSEAS Transactions on Circuits and Systems*, vol. 10, pp. 221-230, 2011.
[34] C. Jeong and S.M. Nowick, "Block-level relaxation for timing-robust asynchronous circuits based on eager evaluation", In Proceedings of the 14th IEEE International Symposium on Asynchronous Circuits and Systems, 2008, pp. 95-104.
[35] P. Balasubramanian, K. Prasad and N.E. Mastorakis, "Robust asynchronous implementation of Boolean functions on the basis of duality," In Proceedings of the 14th WSEAS International Conference on Circuits, 2010, pp. 37-43.
[36] P. Balasubramanian, R. Arisaka and H.R. Arabnia, "RB_DSOP: a rule based disjoint sum of products synthesis method", In Proceedings of the 12th International Conference on Computer Design, 2012, pp. 39-43.
[37] P. Balasubramanian and D.A. Edwards, "Self-timed realization of combinational logic", In Proceedings of the 19th International Workshop on Logic and Synthesis, 2010, pp. 55-62.
[38] P. Balasubramanian, "Self-timed logic and the design of self-timed adders", PhD thesis, School of Computer Science, The University of Manchester, 2010.
[39] P. Balasubramanian and N.E. Mastorakis, "A set theory based method to derive network reliability expressions of complex system topologies," In Proceedings of the Applied Computing Conference, 2010, pp. 108-114.
[40] J. Cortadella, A. Kondratyev, L. Lavagno and C. Sotiriou, "Coping with the variability of combinational logic delays," In Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers and Processors, 2004, pp. 505-508.

[41] V.I. Varshavsky (Ed.), *Self-Timed Control of Concurrent Processes: The Design of Aperiodic Logical Circuits in Computers and Discrete Systems*, Chapter 4: Aperiodic Circuits, pp. 77-85, (Translated from the Russian by A.V. Yakovlev), Kluwer Academic Publishers, 1990.

[42] P. Balasubramanian and K. Prasad, "Early output hybrid input encoded asynchronous full adder and relative-timed ripple carry adder," In Proceedings of the 14th International Conference on Embedded Systems, Cyber-physical Systems, and Applications, 2016, pp. 62-65.

[43] P. Balasubramanian and S. Yamashita, "Area/latency optimized early output asynchronous full adders and relative-timed ripple carry adders," *SpringerPlus*, vol. 5, pages 26, 2016.

[44] P. Balasubramanian and K. Prasad, "Latency optimized asynchronous early output ripple carry adder based on delay-insensitive dual-rail data encoding," *International Journal of Circuits, Systems and Signal Processing*, vol. 11, pp. 65-74, 2017.

[45] K.S. Stevens, R. Ginosar and S. Rotem, "Relative timing," *IEEE Transactions on VLSI Systems*, vol. 11, pp. 129-140, 2003.

[46] D. Bhadra and K.S. Stevens, "Design of a low power, relative timing based asynchronous MSP430 processor," In Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, pp. 794-799, 2017.

[47] M.M. Mano and M.D. Ciletti, *Digital Design*, 4th edition, Prentice-Hall, New Jersey, USA, 2007.

[48] Synopsys Digital Standard Cell Library SAED_EDK32/28_CORE Databook, Revision 1.0.0, 2012.

[49] N.P. Singh, "A design methodology for self-timed systems," MSc dissertation, Massachusetts Institute of Technology, USA, 1981.

[50] W.B. Toms, "Synthesis of quasi-delay-insensitive datapath circuits", PhD thesis, School of Computer Science, The University of Manchester, UK, 2006.

[51] P. Balasubramanian, "A latency optimized biased implementation style weak-indication self-timed full adder," *Facta Universitatis, Series: Electronics and Energetics*, vol. 28, pp. 657-671, 2015.

[52] J. Sparsø and J. Staunstrup, "Delay-insensitive multi-ring structures", *Integration, the VLSI Journal*, vol. 15, pp. 313-340, 1993.

[53] B. Folco, V. Bregier, L. Fesquet and M. Renaudin, "Technology mapping for area optimized quasi delay insensitive circuits", In Proceedings of the IFIP 13th International Conference on Very Large Scale Integration of System-on-Chip, 2005, pp. 146-151.

[54] W.B. Toms and D.A. Edwards, "A complete synthesis method for block-level relaxation in self-timed datapaths," In Proceedings of the 10th International Conference on Application of Concurrency to System Design, 2010, pp. 24-34.

[55] P. Balasubramanian and D.A. Edwards, "A delay efficient robust self-timed full adder", In Proceedings of the IEEE 3rd International Design and Test Workshop, 2008, pp. 129-134.

[56] P. Balasubramanian, D.A. Edwards and W.B. Toms, "Self-timed section-carry based carry lookahead adders and the concept of alias logic," *Journal of Circuits, Systems, and Computers*, vol. 22, pp. 1350028-1–1350028-24, 2013.

[57] P. Balasubramanian, D. Dhivyaa, J.P. Jayakirthika, P. Kaviyarasi and K. Prasad, "Low power self-timed carry lookahead adders," In Proceedings of the 56th IEEE International Midwest Symposium on Circuits and Systems, 2013, pp. 457-460.

[58] P. Balasubramanian, "Asynchronous carry select adders," *Engineering Science and Technology, an International Journal*, vol. 20, pp. 1066-1074, 2017.

[59] P. Balasubramanian and N.E. Mastorakis, "QDI decomposed DIMS method featuring homogeneous/ heterogeneous data encoding", In Proceedings of the International Conference on Computers, Digital Communications and Computing, 2011, pp. 93-101.

[60] P. Balasubramanian and D.A. Edwards, "Power, delay and area efficient self-timed multiplexer and demultiplexer designs," In Proceedings of the 4th IEEE International Conference on Design and Technology of Integrated Systems in Nanoscale Era, 2009, pp. 173-178, 2009.

[61] N.H.E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*, 2nd edition, Addison-Wesley Publishing Company, Massachusetts, USA, 1993.